IN THE UNITED STATES PATENT & TRADEMARK OFFICE

In re application of Heng Chu et al. November 22, 2007

Serial No.: 10/626,340

Filed: July 24, 2003

For: Applying Abstraction to Object Markup Definitions

Art Unit: 2167 Examiner: Kimberly M. Lovel

APPELLANTS' BRIEF ON APPEAL

Mail Stop Appeal Brief - Patents
Commissioner for Patents
P. O. Box 1450
Alexandria, VA 22313-1450

Sir:

This is an Appeal seeking reversal of the decision of the Primary Examiner, finally rejecting all current claims of the subject patent application.

1) REAL PARTY IN INTEREST

The real party in interest is the Assignee, International Business Machines Corporation ("IBM").

2) RELATED APPEALS AND INTERFERENCES

Appellants, the Appellants' legal representative, and the assignee, have no personal knowledge of any other appeals or interferences which will directly affect or be directly affected by or have a bearing on the Board's decision in the pending appeal.

3) STATUS OF CLAIMS

Claims 3 - 5, 7 - 9, and 11 - 33 stand rejected. Claims 3 - 5, 7 - 9, and 11 - 33 are under appeal.

4) STATUS OF AMENDMENTS

An Amendment After Final was filed on September 5, 2007, responsive to the Final Rejection mailed on July 12, 2007. The Amendment After Final was entered by the Examiner, according to the Advisory Action dated September 19, 2007.

5) SUMMARY OF CLAIMED SUBJECT MATTER

1. Appellants' claimed subject matter pertains to validating syntax elements of an input (e.g., during parsing of the input) according to a first syntax level and then using this already-validated input to generate output. The output, however, is generated according to a second syntax level. Because (to paraphrase Appellants' claim language) the second syntax level is *different from* the first syntax level, at least one of the already-validated syntax elements is suppressed when

generating the output (and as a result of this suppressing, the input inherently has a different syntax as compared to the output). Paragraphs 2 - 7, below, describe Appellants' claimed subject matter in more detail, including reference to an example. Paragraphs 8 - 16, below, discuss the claimed subject matter with reference to the specific claim language of Appellants' independent claims. (Citations to Appellants' Specification are provided below in paragraphs 2 - 7 as well as in paragraphs 8 - 16.)

2. For ease of reference, the term "schema" is used herein to refer to a syntax specification. As an example of operation of the claimed invention, a "base" schema may specify a definition of valid syntax for a "person" element, indicating that valid sub-elements in this syntax are "name" and "address" and that valid attributes of a person object conforming to the "person" element are "height" and "weight". Specification, p. 4, lines 2 - 9. See **Fig. 1** for a sample schema providing this specification of a base schema, and **Fig. 2** for an example of an input document that adheres to this base schema specification. Specification, p. 4, lines 10 - 11. The base schema might be extended, whereby the "person" element is extended (i.e., redefined; Specification, p. 4, line 19) to include additional attributes that are deemed useful for a particular application program. Specification, p. 3, lines 15 - 17. For example, an application using "person" elements might find it desirable to extend the schema to include a "gender" attribute (see **Figs. 3A** and **4A**, providing such extended schema and corresponding document, respectively) or an "age" attribute (see **Figs. 3B** and **4B**) or perhaps a "marital status" attribute (**see Figs. 3C** and **4C**). Specification, p. 5, line 2 - p. 6, line 5; p. 7, line 21 - p. 8, line 6. Or, some combination of these additional attributes might be used in a single extended schema (as illustrated in **Figs. 6** and **7**, where **Fig. 6** provides a

schema extension specifying all of these attributes and **Fig. 7** provides a document adhering to this extended schema). Specification, p. 9, lines 1 - 12.

3. If an input document is created according to (i.e., to conform or adhere to) an extended schema, but is parsed according to the base (i.e., unextended) schema, this may generate parsing errors because (by definition) the base schema does not "know about" all of the syntax elements which are specified in the extended schema. (In fact, parsing errors will be generated in this scenario, except in cases where all of the extensions are specified as optional and the adhering input document chooses not to provide any of those optional syntax elements.) Referring again to the example "person" element, if an input document specifies attributes including gender, age, and marital status but this document is parsed according to the base schema that lacks these attributes, each of these 3 attributes will be considered invalid during the parsing. Specification, p. 8, lines 7 - 12; p. 9, lines 14 - 16.

4. In the prior art, this situation is dealt with by (1) turning off the validation of the input (Specification, p. 10, lines 11 - 12; p. 18, lines 1 - 2) or (2) writing customized code to deal with unexpected attributes or elements (Specification, p. 18, lines 2 - 4). Both of these approaches are undesirable: the first because real syntax errors in the input will go undetected (Specification, p. 10, lines 13 - 14), and the second because of the added development expense.

5. The present invention, by contrast, uses two different specifications of valid syntax. The first is used to parse the input, and the second is used to generate output from the parsing. The

need for, and use of, the second (i.e., different) syntax specification will now be described.

6. Continuing with the example scenario, a document generated according to an extended schema might subsequently be provided as input to an application that was not written to use these same extensions. Specification, p. 9, lines 16 - 20. So, suppose that the input document specifies attributes including gender, age, and marital status, and that this input document passes the validation process (because it is validated using an extended schema of the form shown in **Fig. 6**, where all of these attributes are specified), but that this input document is being sent to an application program that uses an extended schema including only the "age" attribute. This is illustrated in **Fig. 10**, where parser **920** receives an input document conforming to a base schema **900** that has been extended (in this illustration, using 3 separate schema extensions **1010**, **910**, **1020** that each provide one of the extended attributes), and this input document is to be provided to "Consumer 2" that only knows about the "age" extension; see reference number **1031**, referring to the extended schema specifying the "age" attribute as "Ext 2". Specification, p. 19, line 9 - p. 20, line 1. If the input document with all 3 of its extended attributes is passed to this application, the application will likely fail – or at least detect what it thinks are errors in the input – because the application does not expect a "person" element to contain the "gender" or "marital status" attributes. Specification, p. 8, lines 13 - 20; p. 9, line 16 - p. 10, line 5 (where the term "most-specific schema extension" refers to the schema that specifies the "superset", or union, of the various extended attributes).

7. To avoid this undesirable result, Appellants' claimed invention generates output using a

schema definition that is different from the schema used for validating the input, and suppresses (from this generated output) those syntax elements which are not valid according to this different schema definition. The "different" schema, for example, is the schema that the receiving application expects its input to adhere to. So, continuing with the example discussing "Consumer 2" of **Fig. 10**, the "gender" and "marital status" attributes (which were, in fact, valid syntax, according to the extended schema with which the document was validated) would be suppressed from the generated output, while the "age" attribute (which is valid according to both the extended schema used for the validating, and the different, also-extended schema referenced at **1031** as being used for the generating) would not be suppressed – and the receiving application will consider this generated output, with its "age" attribute, to be valid syntax (and thus the receiving application will not encounter syntax errors). Specification, p. 16, lines 2 - 14; p. 17, line 17 - p. 18, line 1 (where the term "selected level" is used to refer to the syntax specification used when generating the output and the term "extension level that may be more restrictive" is used to refer to the syntax specification used for validating the input, where "more restrictive" refers to specifying more extensions).

8. Appellants' application includes 5 independent claims, namely Claims 13, 24, 26, 31, and 32. Of these, Claims 13, 31, and 32 are method claims; Claim 24 is a system claim; and Claim 26 is a computer program product claim. The claim language differs among the independent claims, but each captures the above-described notion of a first syntax definition used for validating an input, and a second syntax definition used for generating output from that (already-validated) input; each of these independent claims also recites that at least one of the (already-validated)

syntax elements is suppressed during the generating of the output. The suppressed syntax elements are those that would be invalid according to the second syntax definition: in the example as discussed above in paragraph 7, for example, the suppressed syntax elements would comprise the "gender" attribute and the "marital status" attribute (because those attributes are not known to the second syntax definition, and are therefore considered invalid in view of that second syntax definition). The claim language of each of the independent claims will now be discussed.

9. Independent Claim 13 is a method claim reciting "validating syntax elements of an input ... according to a *first* syntax level while generating output objects, from the [now-validated] input ..., according to a *second* syntax level, wherein the generating further comprises suppressing ... at least one of the validated [i.e., valid according to the first syntax level] syntax elements from the generated output objects in order that the generated output objects will be valid according to the *second* syntax level" (Claim 13, lines 2 - 6, emphasis added; Specification, 22, lines 1 - 11 – noting that the "second syntax level" is referred to therein as "the abstraction level", as defined at p. 17, lines 11 - 12); and "providing the generated output objects [i.e., valid according to the second syntax level] ... for use by an application program" (Claim 13, lines 7 - 8, emphasis added; Specification, p. 21, lines 18 - 19; p. 22, lines 9 - 11). Specification, p. 21, lines 13 - 19 also discuss the suppressing, referring (in the example therein) to generating output using a base schema (i.e., "second syntax level", in Appellants' claim terminology) that does not include any extensions, and therefore suppressing from the output all of the "gender", "age", and "marital status" attributes from the validated source (i.e., input) document.

10. Independent Claim 24 is a system claim reciting "a validating parser usable by a computer" (Claim 24, line 2; Abstract, line 1; Specification, p. 15, line 16); "first means for using the validating parser ... to validate syntax elements specified in an input document ..., wherein the validation is performed according to a *first* syntax level" (Claim 24, lines 3 - 5, emphasis added; Specification, p. 17, lines 18 - 19; p. 18, line 13); and "second means for using the validating parser ... to apply abstraction ... when generating, from the validated [i.e., valid according to the first syntax level] syntax elements, output syntax ... wherein the applying of the abstraction further comprises suppressing, ... from the generated output syntax, at least one of the validated [i.e., valid according to the first syntax level] syntax elements, in order that the generated output syntax ... will be valid according to a *second* syntax level and wherein each of the suppressed syntax elements is valid according to the *first* syntax level but is not valid according to the *second* syntax level" (Claim 24, lines 6 - 13, emphasis added; Specification, p. 17, lines 11 - 17, where the term "selected abstraction level" refers to a syntax definition for the output, termed the "second syntax level" in Appellants' claim language).

11. Independent Claim 26 is a computer program product claim reciting computer-readable program code for "validating, by a parser, syntax elements of an input document according to a *first* schema when parsing syntax of the input document" (Claim 26, lines 3 - 4, emphasis added; Specification, p. 21, lines 15 - 17; p. 22, lines 1 - 6) and "... suppressing ... at least one of the validated [i.e., valid according to the first syntax level] syntax elements when generating output from the parsed syntax of the input document, wherein each of the suppressed syntax elements is valid according to the *first* schema but is not valid according to a *second* schema for which the

output is generated" (Claim 26, lines 5 - 8, emphasis added; Specification, p. 21, lines 18 - 19; p. 22, lines 6 - 11).

12. Independent Claim 31 is a method claim reciting "providing a validating parser that enables a client to dynamically select a syntax abstraction level for use when generating output from the validating parser" (Claim 31, lines 3 - 4, emphasis added; Specification, p. 22, lines 7 - 8); "obtaining an input document ..." (Claim 31, line 5; Specification, p. 22, lines 1 - 3); "validating syntax elements of the input document ... according to a *first* syntax level to which the syntax elements of the input document are to adhere" (Claim 31, lines 6 - 8, emphasis added; Specification 22, lines 4 - 5); and "suppressing at least one of the validated [i.e., valid according to the first syntax level] syntax elements when generating output from the input document ..., wherein: the generated output has syntax that conforms to the syntax abstraction level that has been dynamically selected by the client; the syntax abstraction level is a less-restrictive version [e.g., having in common with the first syntax level an "age" attribute, but not having a "gender" attribute or a "marital status" attribute] of the first syntax level; and each of the suppressed syntax elements is valid according to the *first* syntax level but is not valid according to the [less-restrictive] syntax abstraction level" (Claim 31, lines 9 - 16, emphasis added; Specification, p. 22, lines 6 - 11). In this claim, the "second" syntax is referred to as the "syntax abstraction level". See also p. 17, lines 11 - 14, stating that the term "desired extension level" is equivalent to "desired abstraction level" and referring to the generated "events" (i.e., generated "output", in Appellants' claim terminology) as being "at the selected abstraction level".

13. Finally, independent Claim 32 is a method claim reciting "using ... a *first* syntax level for validating syntax elements when parsing syntax of an input document" (Claim 32, lines 3 - 4, emphasis added; Specification, p. 21, lines 15 - 17; p. 22, lines 1 - 5) and "omitting ... at least one of the validated [i.e., valid according to the first syntax level] syntax elements when generating output ..., wherein each of the omitted syntax elements is valid according to the *first* syntax level [with which it was validated] but is not valid according to a *second* syntax level for which the output is generated" (Claim 32, lines 5 - 8, emphasis added; Specification, p. 21, lines 18 - 19; p. 22, lines 6 - 11).

14. For all of these independent claims, see also **Fig. 9**, and corresponding text on p. 18, line 8 - p. 19, line 8, describing another example where "the parser will validate the full syntax [of the input] according to the extended schema" [i.e., "first" syntax level] (Specification, p. 18, line 13), but instead of using this same syntax definition or schema when generating the output, the output may be generated using an unextended or "base" schema, as described for Consumer 1 and Consumer 3 (Specification, p. 18, lines 15 - 17). The extended schema, in **Fig. 9**, is referred to as "Ext 2" and is shown at **910** as an extension to base schema **900**, where this as-extended schema is used by parser **920** for validating input. Note that Consumer 2, in this example, chooses to have its documents generated with this same extended schema that was used for validating the input (see "Ext 2" at **931**); accordingly, for Consumer 2, there is no "suppressing" or "omitting" of syntax elements (and this scenario is outside the scope of Appellants' claimed invention). However, for Consumer 1 and Consumer 3, which have chosen to receive documents generated according to the base schema (i.e., "second" syntax level, according to Appellants' claim

language), as shown by the notation "Base" at **930** and **932**, the "suppressing" or "omitting" as claimed by Appellants will cause all of the syntax elements which are defined in the "Ext 2" schema extension to be suppressed, or omitted, from the validated input before it is forwarded to that consumer from parser **920**.

15. Independent Claim 24 includes means plus function terminology. Structure, material, or acts supporting this terminology are described in Appellants' Specification, as will now be described.

16. The "validating parser" recited on line 2 of Claim 24 is described at Abstract, line 1; Specification, p. 15, lines 16 - 17; p. 17, lines 11 - 17 (referring to "event-based" parsers as well as "DOM parsers"). For the "first means for using the validating parser ... to validate syntax elements ..." recited on lines 3 - 5 of Claim 24, see Specification, p. 17, lines 18 - 19 and p. 18, line 13. For the "second means for using the validating parser ... to apply abstraction ... when generating ... output syntax ..." recited on lines 6 - 13 of Claim 24, see Specification, p. 17, lines 11 - 17. See also the discussions of processing/computing devices at p. 24, line 18 - p. 26, line 6. Page 25, lines 5 - 15 (and in particular, lines 10 - 15) discuss "... means for implementing the functions specified in the flowchart ..."; see also p. 25, lines 16 - 20, discussing "... instruction means which implement the function specified in the flowchart ...".

6) GROUND OF REJECTION TO BE REVIEWED ON APPEAL

17. The **Ground of Rejection** presented for review is a rejection of Claims 3 - 5, 7 - 9, and

11 - 33 under 35 U.S.C. §103(a) as being unpatentable over U. S. Patent 6,591,260 to Schwarzhoff in view of U. S. Patent Publication 2004/0083221 [stated in the Office Action dated July 12, 2007, hereinafter, "the Office Action", as 2004/00883221] to Dapp et al. (hereinafter, "Dapp").

7) ARGUMENT

18. Paragraph 7 of the Office Action states that Claims 3 - 5, 7 - 9, and 11 - 33 under 35 U.S.C. §103(a) as being unpatentable over U. S. Patent 6,591,260 to Schwarzhoff in view of U. S. Patent Publication 2004/0083221. Of these, the independent claims are Claims 13, 24, 26, 31, and 32.

19. Appellants respectfully submit that a *prima facie* case of obviousness under 35 U.S.C. §103 has not been made out as to these claims, as the cited references, whether taken singly or in combination, do not teach or suggest all the claim limitations.

20. Furthermore, Appellants are entitled to have all words of the claimed invention considered when determining patentability. See Section 2143.03 of the MPEP, "All Claim Limitations Must Be Considered", referencing ***In re Wilson***, 165 USPQ 494, 496 (C.C.P.A. 1970), which stated "*All words* in a claim must be considered in judging the patentability of that claim against the prior art." (emphasis added).

7.1) Rejection of Independent Claims 13, 24, 26, 31, and 32

21. Referring first to independent Claim 32, paragraph 7 of the Office Action cites col. 5, line 19 - col. 6, line 43 of Schwarzhoff as teaching the "using, by a validating parser ..." element recited on lines 3 - 4 of this claim. This same portion of Schwarzhoff is also cited as teaching the "omitting ..." element on lines 5 - 8 of Claim 32 (Office Action, p. 3).

22. Appellants respectfully note that the Office Action states, when discussing the "omitting" claim element, that Schwarzhoff does not teach a "validating parser" ("However, Schwarzhoff fails to explicitly disclose ... wherein the parser is a validating parser."; Office Action, p. 3, next-to-last sentence). Appellants therefore question how Schwarzhoff can be cited for the "using, by a validating parser, a first syntax level for validating syntax elements ..." limitation from lines 3 - 4 of Claim 32. This is a moot point, however, as Appellants respectfully submit that the cited portions of Schwarzhoff cannot be equated to their claim language, as will be demonstrated.

23. The Office Action then cites Dapp as teaching "parsing and validating an XML document using a validating parser", referring to para. **[0026]** of Dapp.

24. Appellants respectfully submit that the cited text in col. 5, line 19 - col. 6, line 43 pertains to using an extended schema for parsing an input document instance (where this parsing might also include *validating*), with no teaching or any suggestion of using a different schema when generating output from the parsed input document. That is, Appellants find no discussion, or suggestion, that can be correlated to their "second syntax level" as recited in Claim 32 ("... wherein each of the omitted syntax elements is valid according to the first syntax level [used for

parsing syntax of an input document; Claim 32, lines 3 - 4] but is not valid according to a second syntax level for which the output is generated" (Claim 32, lines 5 - 8, emphasis added). Instead, Schwarzhoff appears to be silent on the generated output, thereby implying that it uses the same extended schema that was used to parse the input document instance. Problems that may result when using this approach have been discussed in Appellants' specification. See, for example, p. 8, lines 13 - 20 and p. 9, line 14 - p. 10, line 5 (for example, an application may receive attributes for which it has no processing logic; Appellants' specification refers to these as "unexpected" input or "extra" attributes).

25. Appellants acknowledge that the cited text from Schwarzhoff discusses schema extensions. At col. 5, lines 52 - 58, Schwarzhoff discusses extending an "Address" element to contain telephone numbers. Accordingly, a new schema "ContactAddress.sox" is created (col. 5, lines 59 - 62). See also the syntax at reference number **204** of Schwarzhoff's **Fig. 2**, where this phone number extension is specified.

26. However, rather than using two *different* schemas (one for validating input, and one for generating output) as claimed by Appellants, Schwarzhoff describes a technique whereby all "trading partners" (i.e., potential receivers of a document) can dynamically retrieve the *same* extended schema and therefore parse a document created according to that extended schema without parsing errors. See col. 6, lines 20 - 22, "The schemas must be available in a generally available repository to enable trading partners to retrieve them dynamically.", and col. 6, lines 25 - 30, explaining that the trading partner will dynamically locate the schemata "required to correctly

parse" an extended document instance and will then follow links in import statements "to dynamically load [that] new [i.e., extended] schemata". In other words, if a document was created using an extended schema, then Schwarzhoff uses that same extended schema when providing the document to a recipient trading partner, and this use of a single schema is in sharp contrast to Appellants' claimed approach where a first and second schema are used, and where syntax elements are suppressed, or omitted, which are valid according to the first (validating) syntax level but which are not valid according to the second syntax level (e.g., suppressing syntax elements that were introduced by the extension but which are not understood by the application that is receiving the document, as has been described above with reference to the "person" element).

27. Accordingly, Appellants respectfully submit that the cited text of Schwarzhoff does not teach the "second syntax level" as claimed in their claim language. In addition, Appellants find no discussion or suggestion in Schwarzhoff that can be correlated to the "omitting" recited in lines 5 - 8 of Claim 32.

28. The Advisory Action dated September 19, 2007 (hereinafter, "the Advisory Action") presents two numbered paragraphs on the Continuation Sheet, and Appellants respectfully submit that the comments in these paragraphs have misinterpreted Appellants' claimed invention as well as Appellants' comments in their response dated September 5, 2007. Comments made in these paragraphs of the Advisory Action will now be discussed.

29. Paragraph 1 from the Advisory Action cites col. 4, lines 39 - 41 of Schwarzhoff, stating that this text teaches "the XML document has to conform to the referenced schema. Therefore, if the document references an extended schema, the document has to adhere to that particular schema.".

30. While lines 39 - 41 of col. 4 discuss validating a document instance "at two levels", this is a reference to performing two types of validation on an input document. Namely, the document is validated to ensure that it is "well-formed" according to standard XML syntax (for example, ensuring that there are no unclosed angle brackets such as "<document" without a closing ">", and that such element also has a matching "</document>" element) and that it is "valid" according to its particular schema (e.g., ensuring that a purchase order document includes an <Address> element and so forth) . See col. 4, lines 41 - 55, where these two types of validation are discussed in more detail.

31. With regard to the statement from the Advisory Action that "Therefore, if the document references an extended schema, the document has to adhere to that particular schema.", Appellants acknowledge that this is a true statement regarding prior art techniques for processing documents using an extended schema. And, this approach may be used when performing the validating of the input which is recited in Appellants' claim language. However, this prior art approach in no way provides for changing to a different ("second") schema or syntax level, *after* the validation, for generating output from this already-parsed and already-validated syntax from the input document. The prior art approach of using an extended schema referenced by a

document (referring to the statements in paragraph 1 of the Advisory Action) also does not result in omitting any syntax elements, because when using a single referenced schema as in the prior art, the document "adhere[s] to that particular schema" which it references and there is no "second syntax level" involved; accordingly, there is no consideration of whether a syntax element "is valid according to the first syntax level but is not valid according to a second syntax level ..." as recited by Appellants on lines 5 - 8 of Claim 32.

32. Paragraph 2 from the Advisory Action states "If the extended schema fails to include an element, then inherently the element would be suppressed in order to be validated by the schema.". Appellants respectfully disagree with this statement for several reasons. First, if an element is present in a document but not in the schema, then the document is not "validated by the schema". Instead, the parser (if it performs validation) generates an error, and the document is therefore invalidated. Second, this statement from the Advisory Action misstates the approach claimed by Appellants. Appellants are validating the document using the extended schema; therefore, the extended schema will not "fail to include an element". Or stated in another way, the extended schema will include all of the elements from the document. (Note that this statement by Appellants that the extended schema "will include all of the elements ..." assumes that the input document does have valid syntax, according to the extended schema; error cases where a document has invalid syntax are not addressed by Appellants' claim language.) According to Appellants' claim language, it is the second schema (or second syntax level) that might "fail to include an element [or attribute, etc.]". However, this second schema is not being used to validate the input document. Therefore, there is no "inherent suppressing" during validation as

suggested by the Advisory Action.

33. As has been demonstrated above in paragraphs 22 - 32, Schwarzhoff fails to teach or suggest all of the claim limitations of independent Claim 32. Appellants respectfully submit that Dapp fails to provide the missing teachings (and in particular, use of two different syntax levels, one for validating input and the other for generating output, as well as the omitting claimed by Appellants). The references therefore cannot be combined to render Appellants' Claim 32 obvious.

34. Furthermore, as stated above in paragraph 20, Appellants are entitled to have all words of their claim language considered. Paragraphs 22 - 33, above, demonstrate that Schwarzhoff, Dapp, or a combination thereof do not teach, or suggest, all the limitations or all the words specified in independent Claim 32.

35. Accordingly, in view of paragraphs 19 - 34 herein, Appellants respectfully submit that the Office Action fails to make out a *prima facie* case of unpatentability as to independent Claim 32, and without more, this claim is deemed patentable. See ***In re Oetiker***, 24 USPQ 2d 1443, 1444 (Fed. Cir. 1992), which stated:

> If the examination at the initial stage does not produce a prima facie case of unpatentability, then without more the applicant is entitled to grant of the patent.

36. Independent Claims 13, 24, 26, and 31 specify limitations similar to those which have been discussed above with regard to independent Claim 32 (as stated above in paragraph 8), and all of

these claims are rejected in the Office Action using the same references used when rejecting Claim 32. Accordingly, reference is made to paragraphs 19 - 35, above, where the arguments presented therein apply equally to the claim language of independent Claims 13, 24, 26, and 31.

37. Accordingly, Appellants' independent Claims 13, 24, 26, and 31 are deemed patentable over the references.

7.2) Rejection of Dependent Claims 3 - 5, 7 - 9, 11 - 12, 14 - 23, 25, 27 - 30, and 33

38. Dependent Claims 3 - 5, 7 - 9, 11 - 12, 14 - 23, 25, 27 - 30, and 33 stand or fall with independent Claims 13, 24, 26, and 32, from which they depend. Thus, these dependent claims are deemed allowable by virtue of the allowability of independent Claims 13, 24, 26, and 32, the patentability of which is discussed above in "**7.1 Rejection of Independent Claims 13, 24, 26, 31, and 32**".

8) CONCLUSION

For the reasons set out above, Appellants respectfully contend that each appealed claim is patentable, and respectfully request that the Examiner's Final Rejection of appealed Claims 3 - 5, 7 - 9, and 11 - 33 should be reversed.

Respectfully submitted,

/Marcia L. Doubet/

Marcia L. Doubet,
Attorney for Appellants
Reg. No. 40,999

Customer Number for Correspondence: 43168
Phone: 407-343-7586
Fax: 407-343-7587

CLAIMS APPENDIX

CLAIMS AS CURRENTLY PRESENTED:

Claims 1 - 2 (canceled)

Claim 3: The method according to Claim 32, wherein the input document is a structured document.

Claim 4: The method according to Claim 3, wherein the structured document is encoded in Extensible Markup Language ("XML").

Claim 5: The method according to Claim 32, wherein the generated output comprises at least one object representation generated from the input document.

Claim 6 (canceled)

Claim 7: The method according to Claim 33, wherein the second syntax level is requested by specifying a schema name of a schema to which the generated output must adhere.

Claim 8: The method according to Claim 33, wherein the second syntax level is requested by specifying a schema name of a schema to be used by the validating parser when generating the output.

Claim 9: The method according to Claim 8, wherein the schema name is specified, by the application program, as a feature on an invocation of the validating parser.

Claim 10 (canceled)

Claim 11: The method according to Claim 32, wherein the first syntax level is specified in the syntax of the input document.

Claim 12: The method according to Claim 11, wherein the specification in the syntax of the input document uses a schema location construct in the input document.

Claim 13: A computer-implemented method of casting objects, comprising:

validating syntax elements of an input, using a validating parser, according to a first syntax level while generating output objects, from the input using the validating parser, according to a second syntax level, wherein the generating further comprises suppressing, by the validating parser, at least one of the validated syntax elements from the generated output objects in order that the generated output objects will be valid according to the second syntax level; and

providing the generated output objects, by the validating parser, for use by an application program.

Claim 14: The method according to Claim 13, wherein the second syntax level is a less-restrictive version of the first syntax level.

Claim 15: The method according to Claim 13, wherein the first syntax level is a more-restrictive definition of the second syntax level.

Claim 16: The method according to Claim 13, wherein the first syntax level is an extension of the second syntax level.

Claim 17: The method according to Claim 13, wherein the first syntax level represents an extension of the second syntax level.

Claim 18: The method according to Claim 13, wherein the first syntax level and the second syntax level are defined using schemas.

Claim 19: The method according to Claim 18, wherein the schema that defines the first syntax level is an extension of the schema that defines the second syntax level.

Claim 20: The method according to Claim 13, wherein the first syntax level represents a plurality of extensions to the second syntax level.

Claim 21: The method according to Claim 13, wherein the generated output objects adhere to a schema that defines the second syntax level.

Claim 22: The method according to Claim 13, wherein the input adheres to an extended schema that defines the first syntax level.

Claim 23: The method according to Claim 22, wherein the generated output objects adhere to a base schema that is extended by the extended schema.

Claim 24: A system for applying abstraction to object markup definitions, comprising:

a validating parser usable by a computer;

first means for using the validating parser, executing on the computer, to validate syntax elements specified in an input document expressed as an object markup definition, wherein the validation is performed according to a first syntax level; and

second means for using the validating parser, executing on the computer, to apply abstraction to the object markup definition when generating, from the validated syntax elements, output syntax for at least one output object for use by an application program, responsive to the first means, wherein the applying of the abstraction further comprises suppressing, by the validating parser from the generated output syntax, at least one of the validated syntax elements, in order that the generated output syntax of each generated output object will be valid according to a second syntax level and wherein each of the suppressed syntax elements is valid according to the first syntax level but is not valid according to the second syntax level.

Claim 25: The system according to Claim 24, wherein the second syntax level is requested by the application program and wherein the application program then consumes at least one of the at

least one generated output objects.

Claim 26: A computer program product for parsing of input, the computer program product embodied on one or more computer-readable media and comprising:

computer-readable program code for validating, by a parser, syntax elements of an input document according to a first schema when parsing syntax of the input document; and

computer-readable program code for suppressing, by the parser, at least one of the validated syntax elements when generating output from the parsed syntax of the input document, wherein each of the suppressed syntax elements is valid according to the first schema but is not valid according to a second schema for which the output is generated.

Claim 27: The computer program product according to Claim 26, wherein the first schema specifies a first syntax that is a more-restrictive version of a second syntax specified by the second schema.

Claim 28: The computer program product according to Claim 26, wherein the first schema is defined as an extension of the second schema.

Claim 29: The computer program product according to Claim 26, wherein the first schema is defined as an extension of some intermediate schema that extends the second schema.

Claim 30: The computer program product according to Claim 26, wherein the second schema is a

base schema upon which one or more extensions are based, and wherein the first schema is one of the extensions and is based either directly on the base schema or on an intermediate schema that extends the base schema.

Claim 31: A computer-implemented method of providing validation and parsing for clients, comprising:

providing a validating parser that enables a client to dynamically select a syntax abstraction level for use when generating output from the validating parser;

obtaining an input document to be validated and parsed for the client;

validating syntax elements of the input document with the provided validating parser, wherein the validation is performed according to a first syntax level to which the syntax elements of the input document are to adhere; and

suppressing at least one of the validated syntax elements when generating output from the input document with the provided validating parser, for use by the client, wherein:

the generated output has syntax that conforms to the syntax abstraction level that has been dynamically selected by the client;

the syntax abstraction level is a less-restrictive version of the first syntax level; and

each of the suppressed syntax elements is valid according to the first syntax level but is not valid according to the syntax abstraction level.

Claim 32: A computer-implemented method of applying abstraction by a validating parser, comprising:

using, by a validating parser, a first syntax level for validating syntax elements when parsing syntax of an input document; and

omitting, by the validating parser, at least one of the validated syntax elements when generating output from the parsed syntax of the input document, wherein each of the omitted syntax elements is valid according to the first syntax level but is not valid according to a second syntax level for which the output is generated.

Claim 33: The method according to Claim 32, wherein the second syntax level is requested, to the validating parser, by an application program for which the output is generated.

EVIDENCE APPENDIX

Appellants, the Appellants' legal representative, and the assignee have no personal knowledge of evidence requiring separate identification herein as bearing on this Appeal.

RELATED PROCEEDINGS APPENDIX

No related proceedings are personally known to Appellants, the Appellants' legal representative, or the assignee.